

Task Lines and Motion Guides

Paul G. Backes and Stephen F. Peters
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Linh Phan and Kam S. Tso
SoHaRI Incorporated
Beverly hills, California

Abstract

The new task lines and motion guides approaches to telerobotics are described. Motion guides is a new paradigm for teleoperation of a robot where the path is teleoperated rather than the robot, and the robot is constrained to follow the path. Continuous commands to the robot are only one dimensional: forward, back, or halt along the motion guide. Task lines have subtasks attached to motion guides. The task lines and motion guides have been implemented in a virtual reality environment to enable task description and execution to be done in a natural virtual reality graphics environment rather than via direct interaction with a command processor. Subtasks are represented in the virtual reality environment by icons attached to the motion guides. The combination of task lines and motion guides is valuable for ground control of space station robots, which is the initial application for this technology.

1. Introduction

The use of virtual reality graphics environments has enabled the development of important new modes of telerobotics. The first telerobotics systems provided master-slave teleoperation where the motion of a slave robot, at the remote site, follows the motion of a master hand controller moved directly by a human at a local site. These systems were later enhanced with force reflection [1, 2] where the forces between the robot and its environment were transmitted back to the local site operator through the hand controller. Shared control systems enhanced teleoperation by providing a real-time sensory feedback control loop within the remote site robot controller, such as to provide automatic compliance control while the operator was inputting positional commands [3, 4, 2, 5]. Autonomous control enabled the operator at the local site to send commands to the remote robot controller to be executed autonomously, thus eliminating the need for continuous feedback to the operator [6]. Teleprogramming combined features of teleoperation with autonomous control and utilized new graphics technologies by enabling an operator to

generate low level autonomous commands via interaction in a graphical environment and continuously send them for execution on the remote robot, as well as utilizing time delayed sensory feedback to update the calibration of the graphics environment relative to the real environment [7]. A historical perspective of telerobotics can be found in [8].

Requirements for ground control of Space Station robots have motivated the development of two new types of telerobotics, task lines and motion guides. The first constraint is that there will be a round trip communications time delay, on the order of a few seconds, for control of Space Station robots from Earth [9]. It is desired to have continuous input from the operator on the ground for safety purposes and to have the flexibility provided by teleoperation. Also, it is desired to have a system which is not complicated to operate.

The nominal approach to telerobotics for the Space Station has been to use direct teleoperation. The few seconds time delay makes this option very difficult with significant risk. Review displays [10] and shared control can make this safer and more robust, but there remains significant risk.

The Jet Propulsion Laboratory is currently funded by NASA to develop telerobotics technologies to reduce the risks in ground control of space station robots as well as increase ease of use the telerobotic systems for eventual use by the space station program. Task lines and motion guides were developed to meet the needs for ground control of space station robots. Motion guides provide safe teleoperation with time delay. Task lines provide a natural way to specify, execute, and monitor task sequences. The system in which the task lines and motion guide technologies were implemented is described in section 2. Motion guides are described in section 3. Task lines are described in section 4. Reactive sequencing, which allows reactive control within the task lines framework, is described in section 5 followed by conclusions in section 6.

2. System Description

The task lines and motion guides technologies have been developed within the Unified Operator Interface (UOI) which has been developed at JPL to act as the local site operator interface for commanding both manipulators and rovers. An important enabling technology for task lines and motion guides is calibration of the graphical model to the real environment. This graphics calibration technology is provided in the UOI [11]. This enables the operator to have confidence that the virtual reality environment, which the task lines and motion guides are represented in, is a valid representation of the real task environment.

The UOI provides interfaces to a video switcher, a remote site manipulator control system [12], a remote site visual inspection system [13], two computer controlled pan-tilt platforms, one of which also has a computer controlled zoom lens, a six degree-of-freedom (DOF) hand controller, a six DOF spaceball, and a Deneb TeleGRIP graphics environment. The UOI is primarily written in Lisp with additional interface code written in C.

The remote site has a Robotics Research 7 DOF manipulator on a translating rail [12] and a gripper which holds multiple sensors and two cameras.

3. Motion Guides

Motion guides are virtual reality graphical representations of paths that a robot can follow, and the motion guides can be specified or modified via teleoperation. The robot is constrained to following the specified paths. The paths are generated, displayed and modified in a graphical environment. Figures 1, 2, and 3, show a simulated and corresponding real serpentine manipulator following a motion guide into a hole.

3.1 Path Generation

There are various methods for generating a motion guide. One method is to sample the path of a simulated robot which is teleoperated in a graphics environment. Points can be sampled at a temporal or spatial rate and the resulting path can be filtered to smooth it out. This enables the flexibility of a teleoperated system. A motion guide can also be made up of path segments which are each generated separately and given consistent transition conditions. For example, path points could be attached to physical points in

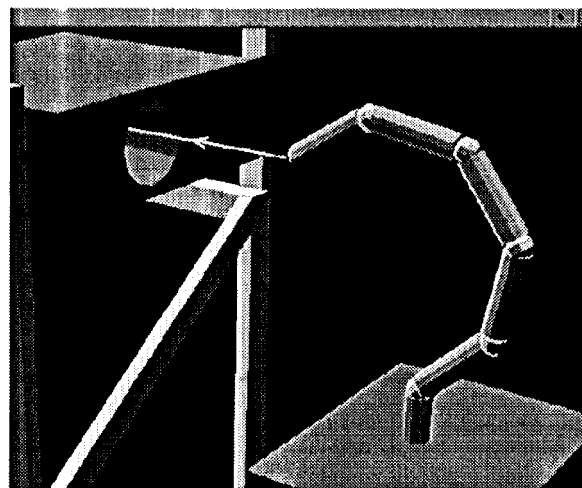


Figure 1: Simulated snake robot at start of motion guide

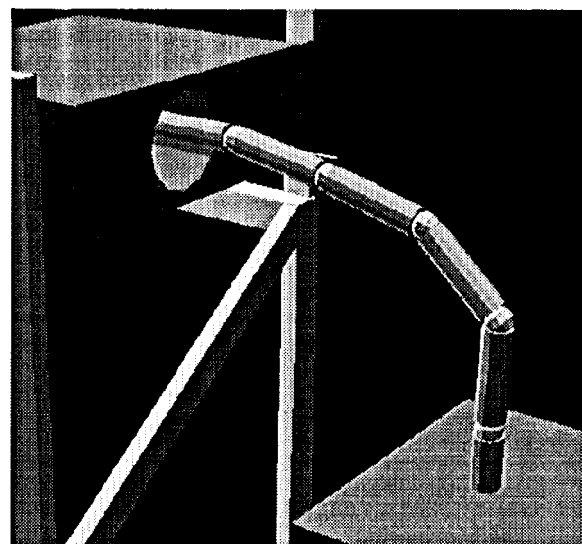


Figure 2: Simulated snake robot at end of motion guide

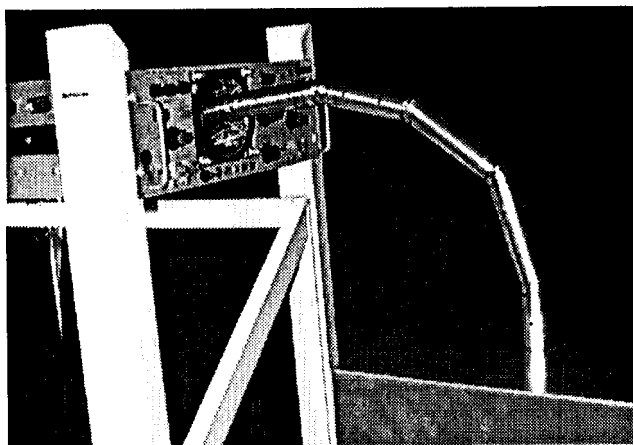


Figure 3: Realsnake robot at end of motion guide inside cavity

the environment such as approach locations to objects or known inspection points. Various other methods can be utilized to generate the paths for motion guides.

The orientational component of the path can be represented in various ways. One option is to draw the robot gripper at various locations along the path. Another option is to thicken the path representation and draw a line on the path which twists as the orientation changes. The approach used in the current implementation is to draw a vector perpendicular to the path at various points along the path. The full six DOF coordinate frame is generated by assuming that one vector is tangential to the path, the second vector is perpendicular to the path and goes through the gripper fingers, and the third vector is generated from the first two.

3.2 Path Modification

A distinguishing feature of motion guides is that the path can be easily modified both before and while the robot is moving along it. For example, for the snake manipulator the operator can point the path into the center of the hole before or while the robot is moving along the motion guide. The robot is constrained to follow the motion guide so when the motion guide is moved by the operator, the graphical robot, and corresponding real robot, move along a minimal normal distance to stay on the motion guide. The operator at the control station teleoperating a motion guide is shown in figure 4.

Figures 5 and 6 show the simulated Robotics Research 7 DOF robot at the beginning and end of a



Figure 4: Operator teleoperating a motion guide

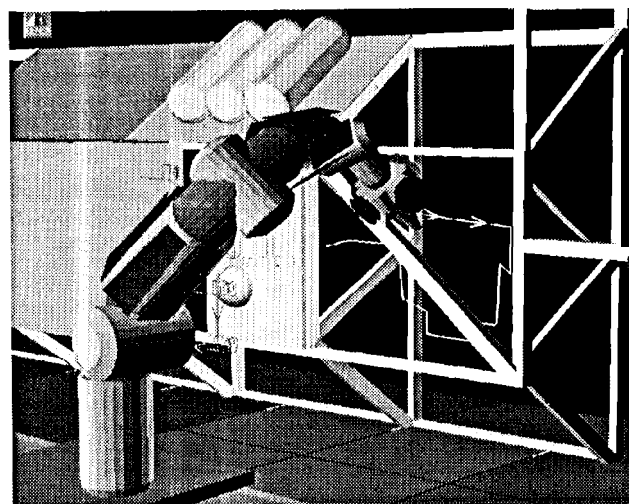


Figure 5: Simulated robot at beginning of motion guide

motion guide which will take it into a truss. Figure 6 shows the simulated robot near the end of the motion guide inside of the truss. The left image is the simulated view from the gripper camera. The operator teleoperates the motion guide to generate a motion guide which provides a collision free path for the robot and such that at the end of the motion guide, the robot can see the ammonia bottle. Figure 7 shows the real robot near the end of the motion guide inspecting the ammonia bottle.

Safe motion along the path can be verified in several ways. The motion along the path can be simulated before having the real robot move along the path to ensure that the path is safe. A planned feature is to have automated collision detection for the robot at all times along whole motion guide, not for just the current sim-

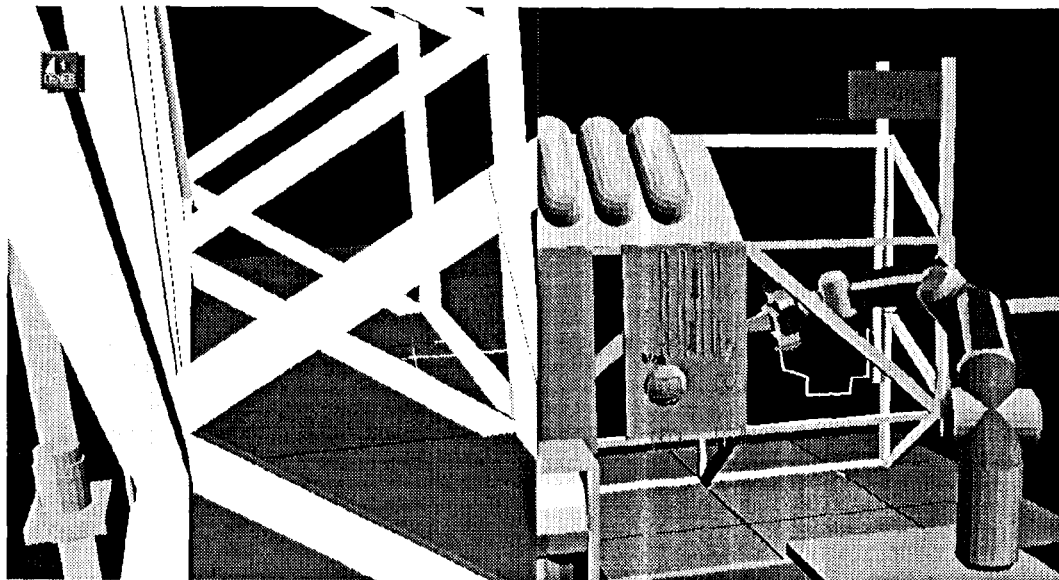


Figure 6: Simulated robot near end of motion guide; left view is from gripper camera viewpoint

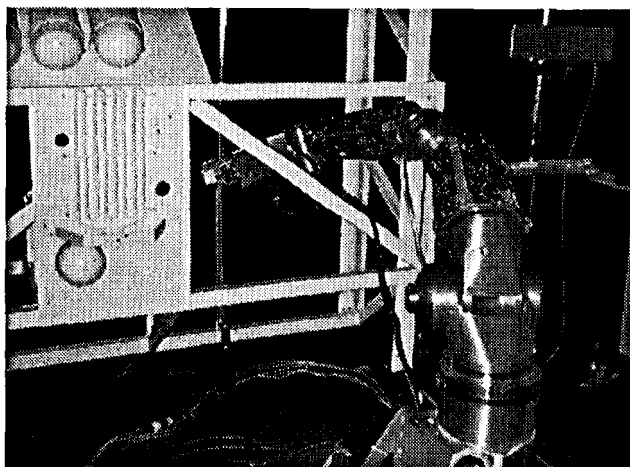


Figure 7: Robot near end of motion guide

ulated position. This will allow the operator to know that a motion guide represents a safe trajectory without separately verifying the path via a separate simulation step. The operator can then generate or modify a path and anywhere a potential collision would occur, the path will be highlighted. The operator will then modify the path to avoid that collision. Another planned feature is automated obstacle avoidance. Here, the path will be autonomously modified to prevent the robot from any collisions. Thus as the operator generates or modifies the motion path, other parts of the path will be assured to not also be modified in such a way as to cause a collision.

3.3 Motion Execution

Another distinguishing feature of motion guides is that the operator "[drives]" the robot down the motion guide. The motion guide provides the spatial path that the robot will follow, but not the time at which to be at any point. The operator therefore inputs one DOF of motion to move the robot forward or backward along the path. In the current implementation, the operator pushes on a spaceball, as shown in figure 8. Pushing forward on the spaceball causes the robot to move forward along the path and pushing backward causes the robot to move back along the path. If the operator does not push on the spaceball, then the motion stops. This driving of the robot along the motion guide is independent of the teleoperation of the motion guide itself, so the operator can switch between teleoperating the



Figure 8: Operator commanding motion along a motion guide

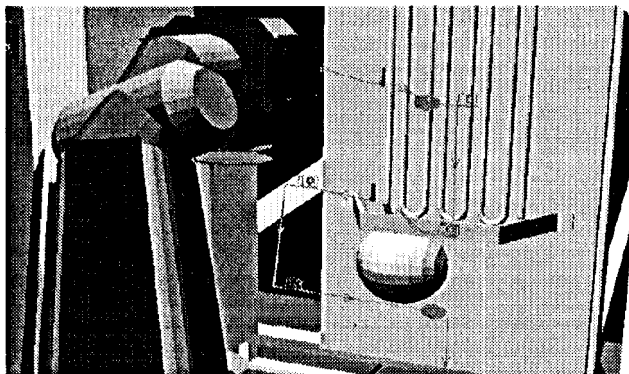


Figure 9: Taskline for visual, temperature, and gas inspection

motion guide and driving the robot down the motion guide.

Another mode is provided to the operator where the operator can command the robot to move along the motion guide at a constant velocity until a new command to halt or change direction is given. With this mode, the robot can be moving along the motion guide while the operator is teleoperating the motion guide or extending the motion guide.

4. Task Lines

Task Lines are visual representations of command sequences where visual icons representing subtasks are attached at different points on motion guides. A task line for multi-sensor inspection is shown in figure 9 with the corresponding real robot shown in figure 10. Automated visual inspection is represented by an icon which looks like a camera. Temperature inspection is represented by an icon which looks like a thermometer. Gas inspection is represented by an icon which looks like a

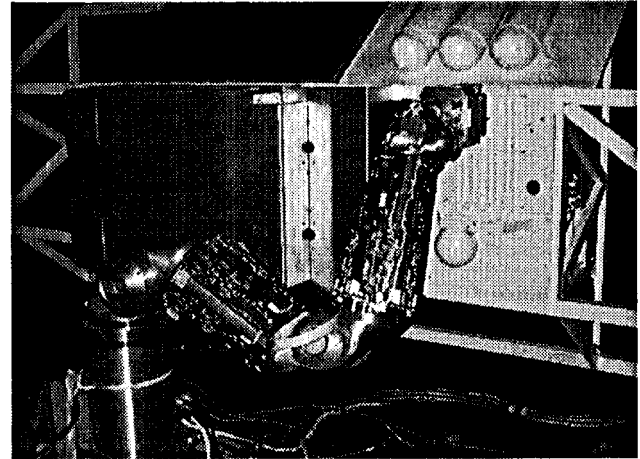


Figure 10: Robot executing taskline

cloud. One goal with task lines is to provide programming within the graphical environment. A list of icons representing various subtasks will be provided within the graphical environment which the operator can select and drag to desired points on the motion guide. The operator will be able to modify the positions of the icons on the motion guides as well as modify the motion guides. The operator will be able to select an icon and then be presented with a panel which describes the associated command or sub-sequence and allow the operator to input parameters or modify the associated subtask. The current graphics environment limits the graphical icons to be wire frame models which limits the visibility of the icons. It is envisioned that icons will have generic shapes for specific types of commands, but then have specific additions to the icon to represent specific parameterization of the associated command.

Figure 11 shows the Unified Operator Interface (UOI) with integrated task lines, motion guides, and command sequence generation and execution. The command sequence shown below the graphics in figure 11 is the command sequence corresponding to the task line. Whenever the command sequence or task line is modified, the corresponding task line or command sequence will be automatically updated to allow the operator to modify the command sequence in the most convenient way. For the laboratory experiment, the operator and ground control station of figure 8 were in a separate building from the robot and task environment shown in figure 10.

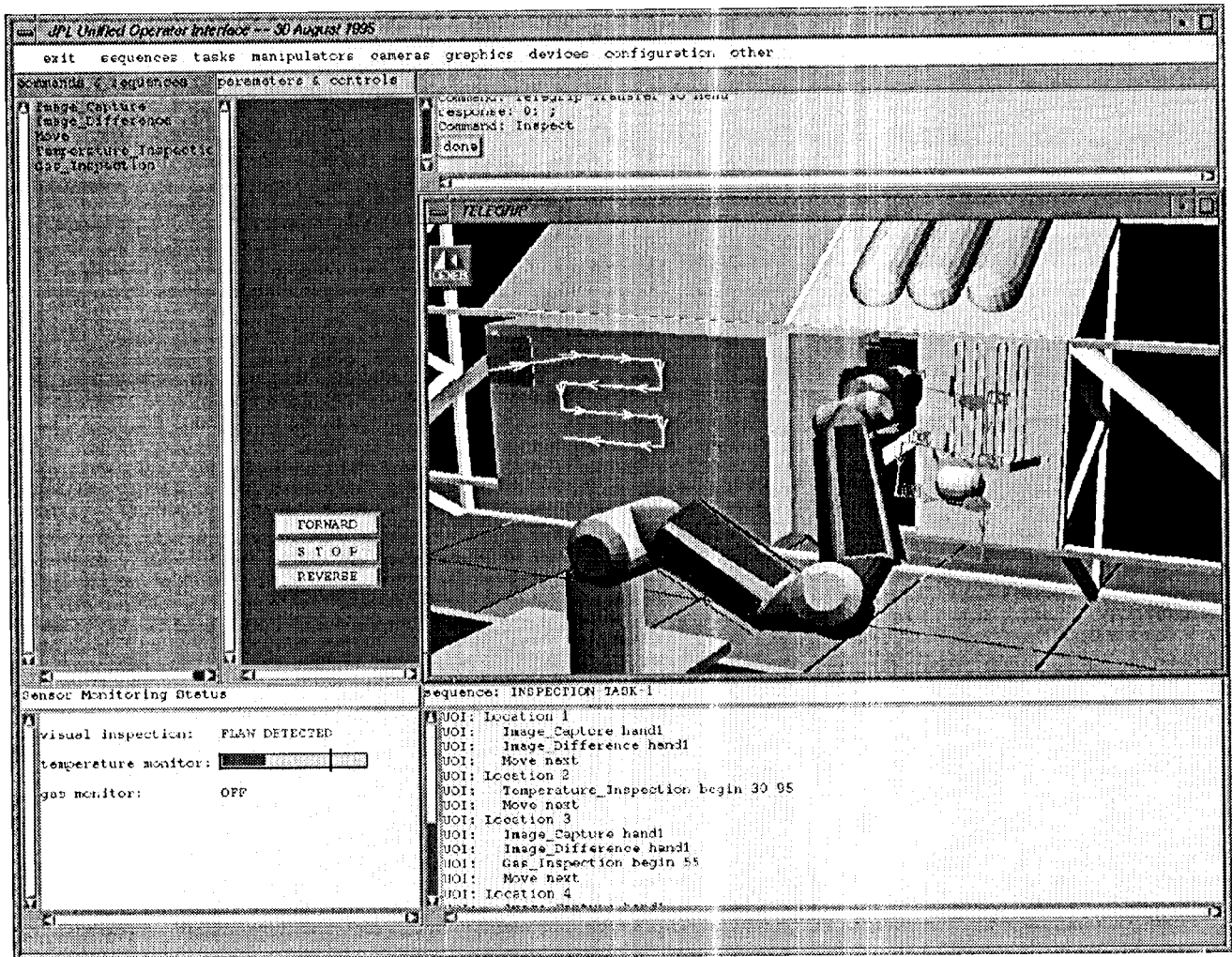


Figure 11: Unified Operator Interface with task line and corresponding command sequence

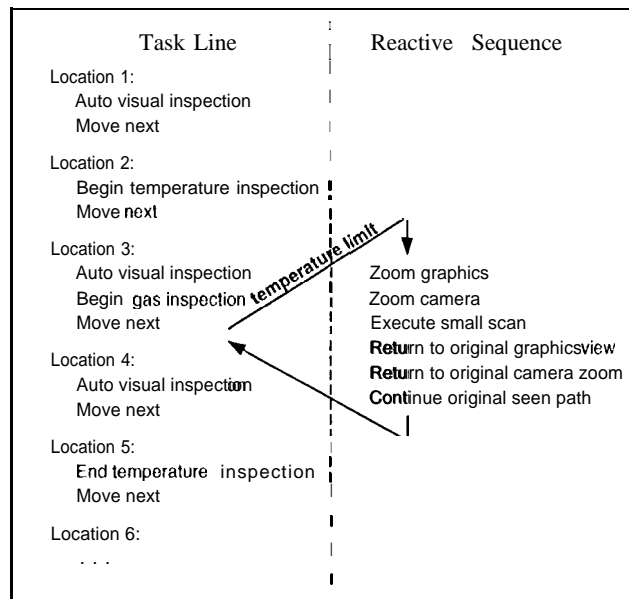


Figure 12: Example of command flow for task line with reactive sub-sequence

5. Reactive Sequencing

Reactive sequencing allows sub-sequences to be automatically executed when a given event occurs. This is depicted in figure 12 for the task line experiment described here. Temperature inspection is started at location 2. Therefore as the robot moves along the task line, the temperature is continuously measured by the point temperature sensor in the gripper. The experiment apparatus included a heat source on the truss panel between locations 3 and 4. As depicted in figure 12, and as actually occurred in the actual experiment, when the robot moved over the hot spot, the system detected that the temperature was above the specified threshold, and the system automatically initiated the sub-sequence described in figure 12. The graphics view automatically was zoomed in the area where the hot spot was detected. The camera was automatically commanded to point and zoom in on the detected hot spot. The robot was automatically commanded to begin a small amplitude, high frequency scan of the area where the hot spot was detected. After the small scan was completed, the system automatically returned the graphics view to its view when the hot spot was detected and commanded the camera to return to its pan-tilt-zoom state that it was in before the hot spot was detected. Then the robot was automatically commanded to continue on its original scan path.

The reactive sequences are associated with events which are specified in the task line commands. In this case the temperature inspection command specified for the system to signal a temperature anomaly event when the measured temperature rises above 95 degrees Fahrenheit. There may be reactive sequences associated with numerous events. Automated selection of graphical views, automated simulation of robot motion, and automated generation of camera pan-tilt-zoom commands are commands within a reactive sequence associated with the "Move next" command. Whenever a "Move next" command is issued, these operations are performed before moving the robot. The system first automatically selects a best graphical view to observe the motion and then simulates the robot motion. Then the best camera is selected and pan-tilt-zoom commands for that camera are selected and sent to the camera. Then the commands are sent to the real robot.

6. Conclusions

The new task lines and motion guides approaches to telerobotics have been described along with their implementation and use in a real telerobotic system. Motion guides is a valuable new method for teleoperation where the path is teleoperated in a graphical environment and the robot is constrained to follow the path, as opposed to traditional teleoperation where the robot is teleoperated directly. Task lines is a valuable new method for command sequence generation and visualization which is more intuitive and easy to use than using command programs directly. These technologies have been developed and demonstrated in a laboratory system. They are planned to be transferred to Johnson Space Center in the coming year to be integrated into International Space Station telerobotics development systems.

Acknowledgements

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

- [1] A. K. Bejczy and J. K. Salisbury. Kinesthetic coupling between operator and remote manipulator. In *Proceedings A SME International Computer Technology Conference*, volume 1, pages 197-211, San Francisco, California, 1980.

- [2] W. S. Kim, B. Hannaford, and A. K. Bejczy. Force-reflection and shared compliant control in operating telemanipulators with time delay. *IEEE Trans. on Robotics and Automation*, 8(2):176-185, 1992.
- [3] S. Hayati and S. '1'. Venkataraman. Design and implementation of a robot control system with traded and shared control capability. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 1310-1315, 1989.
- [4] Paul G. Backes and Kam S. Tso. Umi: An interactive supervisory and shared control system for telerobots. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 1096-1101, Cincinnati, Ohio, May 1990.
- [5] Paul G. Backes. Dual-arm supervisory and shared control task description and execution. *Robotics and Autonomous Systems*, 12:29-54, 1994.
- [6] Paul G. Backes. *Telerobotics and Robotics in Space*, chapter 6: Supervised Autonomy For Space Telerobotics. AAA, 1993.
- [7] Janez Funds, Thomas S. Lindsay, and Richard P. Paul. Teleprogramming: Toward delay-invariant remote manipulation. *Presence*, 1(1):29-44, Winter 1992.
- [8] Thomas Sheridan. *Telerobotics, Automation, and Human Supervisory Control*. M. I.T. Press, 1992.
- [9] R. Aster, J.M. de Pitalaya, and G. Deshpande. Analysis of end-to-end information system latency for space station freedom. Jet Propulsion Laboratory, Internal Document 1-8650, May 1991.
- [10] W. S. Kim and A. K. Bejczy. Demonstration of a high-fidelity predictive/preview display technique for telerobotic servicing in space. *IEEE Trans. on Robotics and Automation*, 9(5):698-702, 1993.
- [11] Won S. Kim. Virtual reality calibration for telerobotic servicing. In *Proceedings IEEE International Conference on Robotics and Automation*, volume 4, pages 2769-2775, 1994.
- [12] David Lim, Thomas S. Lee, and Homayoun Seraji. A real-time control system for a mobile dexterous 7 dof arm. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 1188-1195, San Diego, California, May 8-12 1994.
- [13] J. Balaram and K. V. Prasad. Automated inspection for remote telerobotic operations. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 883-888, 1993.